RESEARCH ARTICLE

FlyLimbTracker: An active contour based approach for leg segment tracking in unmarked, freely behaving *Drosophila*

Virginie Uhlmann[1☯]*, Pavan Ramdya[2,3☯¤a]*, Ricard Delgado-Gonzalo[1¤b], Richard Benton[3], Michael Unser[1]

1 Biomedical Imaging Group, École Polytechnique Fédérale de Lausanne (EPFL), Lausanne, Switzerland, **2** Institute of Microengineering, École Polytechnique Fédérale de Lausanne (EPFL), Lausanne, Switzerland, **3** Center for Integrative Genomics, Faculty of Biology and Medicine, University of Lausanne, Lausanne, Switzerland

☯ These authors contributed equally to this work.
¤a Current address: Division of Biology and Bioengineering, California Institute of Technology, Pasadena, California, United States of America
¤b Current address: Centre Suisse d'Électronique et Microtechnique (CSEM), Neuchâtel, Switzerland
* pavan.ramdya@epfl.ch (PR); virginie.uhlmann@epfl.ch (VU)

Abstract

Understanding the biological underpinnings of movement and action requires the development of tools for quantitative measurements of animal behavior. *Drosophila melanogaster* provides an ideal model for developing such tools: the fly has unparalleled genetic accessibility and depends on a relatively compact nervous system to generate sophisticated limbed behaviors including walking, reaching, grooming, courtship, and boxing. Here we describe a method that uses active contours to semi-automatically track body and leg segments from video image sequences of unmarked, freely behaving *D. melanogaster*. We show that this approach yields a more than 6-fold reduction in user intervention when compared with fully manual annotation and can be used to annotate videos with low spatial or temporal resolution for a variety of locomotor and grooming behaviors. FlyLimbTracker, the software implementation of this method, is open-source and our approach is generalizable. This opens up the possibility of tracking leg movements in other species by modifications of underlying active contour models.

OPEN ACCESS

Citation: Uhlmann V, Ramdya P, Delgado-Gonzalo R, Benton R, Unser M (2017) FlyLimbTracker: An active contour based approach for leg segment tracking in unmarked, freely behaving *Drosophila*. PLoS ONE 12(4): e0173433. https://doi.org/10.1371/journal.pone.0173433

Editor: Giorgio F Gilestro, Imperial College London, UNITED KINGDOM

Received: January 1, 2017

Accepted: February 11, 2017

Published: April 28, 2017

Data Availability Statement: User instructions, FlyLimbTracker software, sample data, and a video demonstration of a complete data analysis pipeline can be found at: http://bigwww.epfl.ch/algorithms/FlyLimbTracker/ and at https://doi.org/10.6084/m9.figshare.4688962.v1.

Funding: This work was supported by Swiss SystemsX.ch Initiative, Grant#: 2008/005, http://www.systemsx.ch/, VU, RDG, MU; Swiss National Science Foundation, Grant#: 200020-121763 and 200020-144355, http://www.snf.ch/en/Pages/

Introduction

Many terrestrial animals rely on complex limb movements to locomote, groom, court, mate, and fight. Discovering how these and other fundamental behaviors are orchestrated by the nervous system will require manipulations of the genome and nervous system as well as quantitative measurements of behavior. The vinegar fly, *Drosophila melanogaster*, is an attractive model organism for uncovering the neural and genetic mechanisms underlying behavior. First, it boasts formidable genetic tools that allow experimenters to remotely activate, silence, visualize and modulate specific gene function in identified neurons [1]. Second, a number of

default.aspx, VU, RDG, MU; Human Frontier Science Program, Grant#: LT000057/2009, http://www.hfsp.org/, PR; Swiss National Science Foundation, Grant#: P300P3_158511, http://www.snf.ch/en/Pages/default.aspx, PR; European Research Council, Grant#: 205202 and 615094, https://erc.europa.eu/, RB. The funders had no role in study design, data collection and analysis, decision to publish, or preparation of the manuscript.

Competing interests: The authors have declared that no competing interests exist.

sophisticated methods have been developed that permit robust tracking and analysis of *D. melanogaster* body movements–a promising set of tools for biological screens [2–9].

By contrast, similarly robust methods with the precision required to semi-automatically track leg segments are largely absent. State-of-the-art approaches suffer from several drawbacks. For example, the most precise methods require the manual placement of visible markers on tethered animals [10] (for another example in cockroaches see [11]) as well as sophisticated fluorescence-based optics. Marking insect leg segments is a time-consuming process that limits experimental throughput. On the other hand, the most high-throughput approach for marker-independent leg tracking in freely behaving *Drosophila* uses complex optics to measure Total-Internal-Reflection Fluorescence (TIRF) when the distal leg tips (claws) of walking animals scatter light transmitted through a transparent floor [12]. An image-processing based method for claw tracking has also been developed [13]. Although these methods can resolve the claws of each leg, they cannot detect their segments. Thus, they provide only binary information about whether or not a leg is touching the surface and cannot resolve the velocity of joints during swing phases, stance adjustments, or non-locomotive limb movements such as reaching [14] or grooming [15]. Ultimately, such measurements will be necessary to gain a complete understanding of how the nervous system controls each limb.

Here we describe a new method that permits semi-automated, marker-free tracking of the body and leg segments of freely walking *Drosophila*. We implement this method in an open source software plug-in named FlyLimbTracker for Icy, an open-source, community-maintained, and user-friendly image processing environment for biological applications [16–18]. Our approach relies purely on image analysis of high-speed, high-resolution videos. Thus, it does not require complicated optical setups like those used for TIRF [12] or fluorescence imaging [10]. Specifically, FlyLimbTracker uses active contours (i.e., snakes) to process objects in high-frame-rate image sequences. There are a number of active contour algorithms [19]; here we use parametric spline-snakes. These global-purpose, semi-automated image segmentation algorithms are typically used in two steps. First, the user roughly initializes a curve to a feature in an image (e.g., a fly's body or leg). Second, the curve's shape is automatically optimized to fit the boundaries of the object of interest. Therefore, segmentation algorithms using spline-snakes are composed of two major components: a *spline curve* or *model* that defines how the snake is represented in the image, and a *snake energy* that dictates how the curve is deformed in the image plane during optimization. Spline-snake models have a number of advantages to other approaches: they are (i) composed of only a few parameters, (ii) very flexible, (iii) amenable to easy manual edits, and (iv) formed from continuously defined curves that permit refined data analysis. Such models have therefore become widely used for image segmentation in biological applications [20,21]. Using this approach, we show that FlyLimbTracker can semi-automatically track freely walking or grooming *D. melanogaster* in video data that spans a wide range of spatial and temporal resolutions. FlyLimbTracker reduces the number of user clicks required–a proxy for annotation speed–by approximately 6-fold (see Results). FlyLimbTracker is written as a plug-in for Icy. This makes it amenable to customization for behavioral measurements in flies with altered morphologies (e.g., following leg removal) and, potentially, in other species (e.g., stick insect, cockroach).

Materials and methods

Drosophila behavior experiments

We performed experiments using adult female *Drosophila melanogaster* of the *Canton-S* strain at 2–4 days post-eclosion. Flies were raised on a 12 h light:12 h dark cycle at 25˚C. Experiments were performed in the late afternoon Zeitgeber time after flies were starved for 4–6 h in humidified 25˚C incubators.

During experiments, we placed flies in a custom designed acrylic arena (pill shaped: 30 mm x 5 mm x 1.2 mm) illuminated by a red ring light (FALCON Illumination MV, Offenau, Germany). We captured behavioral video using a high-speed (236 frames-per-second), high-resolution (2560 x 918 pixels) camera (Gloor Instruments, Uster Switzerland) viewing animals from below.

Automated body and leg tracking

FlyLimbTracker is implemented in Java as a freely available plug-in for Icy, a cross-platform, multi-purpose image processing environment [16]. Briefly, FlyLimbTracker performs leg segment tracking in several steps. First, the user is asked to manually initialize the position of a fly's body and leg segments in a single frame of the image sequence. This information is combined with image features to propagate body and leg segmentation to the frames immediately preceding, or following this first frame. At any time, the user can stop, edit, and restart automated segmentation. Manual corrections are taken into account when tracking is resumed.

To perform image segmentation, FlyLimbTracker uses active contour models (i.e., snakes). A snake [22] is defined as a curve that is optimized from an initial position—usually specified by the user—toward the boundary of an image object. Evolution of the curve's shape results from solving an optimization problem in which a cost function, or snake energy, is minimized. Thus, snakes are an effective hybrid, semi-automated algorithm in which user interactions define an initial position from which automated segmentation proceeds [23,24]. Specifically, FlyLimbTracker first uses a closed snake to segment the *Drosophila* body into a head, thorax, and abdomen. Then, open snakes are used to model each of the fly's legs. Manual mapping of these snakes onto the fly in an initial frame is the basis for subsequent tracking. Hereafter, we formalize and illustrate the construction of segmentation models for the fly's body and legs, respectively. We then describe how these models are propagated to track the fly throughout an image sequence. Finally, we discuss the kind of data that can be acquired using our plug-in, and provide details about software implementation and availability.

***Drosophila* body model.** We designed a custom snake model to segment and track the *Drosophila* body. In our model, the fly's body is defined as a 2-dimensional closed curve **r** composed of *M* control points:

$$\mathbf{r}(\mathrm{t}) = \begin{pmatrix} \mathrm{r}_1(\mathrm{t}) \\ \mathrm{r}_2(\mathrm{t}) \end{pmatrix} = \sum_{\mathrm{k}=0}^{\mathrm{M}-1} \mathbf{c}[\mathrm{k}]\varphi_{\mathrm{M}}(\mathrm{Mt}-\mathrm{k}), \tag{1}$$

with $t \in [0,1]$, where $\mathbf{c}[k] = \{(\mathrm{c}_1[k]\ \mathrm{c}_2[k])^{\mathrm{T}}\}_{k\in\mathbb{Z}}$ is the *M*-periodic sequence of control points and $\varphi_M(t) = \sum_{n=-\infty}^{\infty} \varphi(t - Mn)$ the *M*-periodization of a basis function φ. For a detailed description of the spline snake formalism, see [19]. The proposed model for the body of the fly consists of an $M = 18$ nodes snake using the ellipse-reproducing basis [25]

$$\varphi(t) = \begin{cases} \dfrac{\cos\left(\dfrac{2\pi|t|}{M}\right)\cos\left(\dfrac{\pi}{M}\right) - \cos\left(\dfrac{2\pi}{M}\right)}{1-\cos\left(\dfrac{2\pi}{M}\right)}, & 0 \leq |t| < \dfrac{1}{2}, \\ \dfrac{1-\cos\left(\dfrac{2\pi\left(\dfrac{3}{2}-|t|\right)}{M}\right)}{2\left(1-\cos\left(\dfrac{2\pi}{M}\right)\right)}, & \dfrac{1}{2} \leq |t| < \dfrac{3}{2}, \\ 0, & |t| \geq \dfrac{3}{2}. \end{cases} \tag{2}$$

To optimize the snake automatically from a coarse initial position to the precise boundaries of the fly's body, we define a snake energy composed of three elements:

$$E_{\text{body}} = E_{\text{edge}} + E_{\text{region}} + E_{\text{shape}}. \tag{3}$$

The first element E_{edge} is an edge-based energy term relying on gradient information to detect the body contour, which is formally expressed as

$$E_{\text{edge}} = -\oint_{\mathbf{r}} \mathbf{k}^T (\nabla I(x,y) \times d\mathbf{x}), \tag{4}$$

where $d\mathbf{x}$ is the infinitesimal vector tangent to the snake, $\nabla I(x,y)$ the in-plane gradient of the image at position (x,y), $\mathbf{k} = (0,0,1)$ is the vector orthonormal to the image plane, and $\mathbf{r}$ is the snake curve. The energy term is negative since it has to be minimized during the optimization process. Using Green's theorem, we can transform the line integral into a surface integral:

$$E_{\text{edge}} = -\int_{\Omega} \Delta I(\mathbf{x}) d\mathbf{x}, \tag{5}$$

with Ω the region enclosed by the snake curve, and $\Delta I(\mathbf{x})$ the Laplacian of the image at position $\mathbf{x} = (x,y)$.

The second term, E_{region}, is a region energy term that uses region statistics to segment the object from the background. Specifically, it is computed as the intensity difference between the region enclosed by the snake Ω and the region surrounding it $\Omega_\lambda \backslash \Omega$, as

$$E_{\text{region}} = \frac{1}{|\Omega|}\left(\int_{\Omega} I(\mathbf{x}) d\mathbf{x} - \int_{\Omega_\lambda\ \Omega} I(\mathbf{x}) d\mathbf{x}\right), \tag{6}$$

where I is the image, Ω the region enclosed by the snake curve, and $|\Omega|$ the signed area of the snake, which is defined as

$$|\Omega| = \oint_{r} r_2 dr_1, \tag{7}$$

where r_1, r_2 and r are given by Eq 1.

Minimizing this term encourages the snake to maximize the contrast between the area it encloses and the background. For more details about the edge and region energy derivations, see [26,27].

Finally, the last term, E_{shape}, corresponds to the shape-prior energy contribution detailed in [28]. This term measures the similarity between the snake and its projection on a given reference curve. It therefore encourages the convergence of the contour to an affine transformation of the reference shape. The smoothness and regularity of the reference are preserved. Moreover, this term prevents the formation of loops and aggregation of nodes during the optimization process. In our case, the reference shape is a symmetric 18-node fly body contour (Fig 1A and 1F).

To automatically optimize the snake, we modified the position of the control points by minimizing the energy using a Powell-like line-search method [29], a standard unconstrained optimization algorithm that converges quadratically to an optimal solution. First, one direction is chosen depending on the partial derivatives of the energy. Since the energy is continuously defined, finite differences–a discrete approximation of the continuous derivative–is used to estimate partial derivatives with respect to each of the control points. Second, a one-dimensional minimization of the energy function is performed in the selected direction. Finally, a new direction is chosen using the partial derivatives and enforcing conjugation properties. These steps are repeated until convergence. The final configuration of the control points provides an accurate description of the orientation and size of the fly body.

In practice, the algorithm depends on initial user input to coarsely locate the fly in a frame of the image sequence. Following a single mouse click, a two-step multiscale optimization scheme inspired by [27] is initiated. A spherical active contour composed of 3-control points is first created, centered at the mouse position. This snake is optimized using $E_{\text{edge}} + E_{\text{region}}$ to form an elliptic curve surrounding the fly. In this way, the major axis of the elliptical snake will be aligned with the anteroposterior axis of the fly, and the minor axis will be perpendicular to it.

The 3-point elliptical snake fit to the body of the fly can be expressed as follows [26]:

$$\mathbf{r}(t) = \mathbf{R}_0 + \mathbf{R}_1 \cos(2\pi t) + \mathbf{R}_2 \sin(2\pi t), \tag{8}$$

where $t \in [0,1)$, $k \in \mathbb{Z}$, and

$$\mathbf{R}_0 = \frac{1}{3}\sum_{k=0}^{2}\mathbf{c}[k], \qquad \mathbf{R}_1 = \sum_{k=0}^{2} h_c[k]\mathbf{c}[k], \qquad \mathbf{R}_2 = \sum_{k=0}^{2} h_s[k]\mathbf{c}[k], \tag{9}$$

with

$$h_c[k] = \frac{2}{3}\cos\left(\frac{\pi}{3}\right)\cos\left(\frac{2\pi k}{3}\right), \quad h_s[k] = \frac{2}{3}\cos\left(\frac{\pi}{3}\right)\sin\left(\frac{2\pi k}{3}\right). \tag{10}$$

Note that the $\mathbf{c}[k]$ correspond to the control points of the snake, as in Eq 1.

Relating this to the general parametric equation of an ellipse of major axis a, minor axis b, and center $(x_c\ y_c)^{\mathrm{T}}$ allows us to extract the parameters of the 3-control point snake fit to the fly's body. Namely, $(x_c\ y_c)^{\mathrm{T}} = \mathbf{R}_0$, $a = \max(\|\mathbf{R}_1\|,\|\mathbf{R}_2\|)$ and $b = \min(\|\mathbf{R}_1\|,\|\mathbf{R}_2\|)$. By knowing a, the orientation of the ellipse in the image can be computed.

The ellipse fit is then replaced by an 18-node fly-shaped closed snake that has been rotated and dilated to match the ellipse's length and orientation (Fig 1A). An ambiguity results since two potential snake models can be initialized for a given ellipse, with opposite anteroposterior axis orientation. To resolve this ambiguity, both potential snake orientations are optimized on the image using E_{body} in addition to E_{edge} and E_{region}. The solution with the lowest cost (i.e., energy value at convergence) is used.

***Drosophila* leg model.** Once the fly's body is properly segmented, open snake models for each of its legs are then added. First, the positions of leg coxa-thorax attachment points (hereafter referred to as *anchors*) are automatically computed based on the body segmentation. The location of the six leg anchors with respect to the reference body model have been empirically determined as linear combinations of three axes defined by the head-thorax junction, the

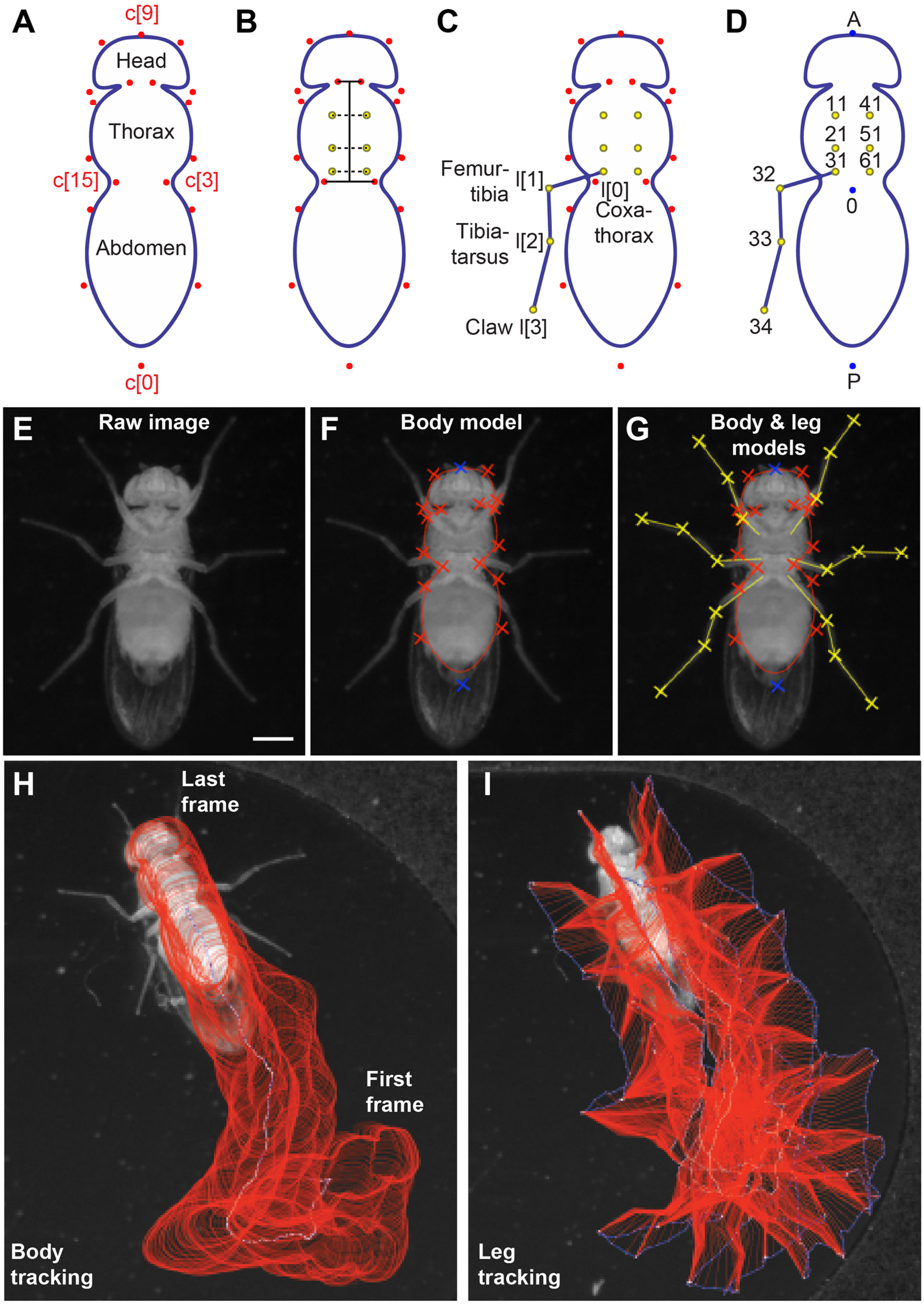

Fig 1. FlyLimbTracker uses active contour models to annotate the *Drosophila* body and legs. (A) The body model is a closed snake consisting of 18 control points (**c**[0] to **c**[17]). Control points **c**[0] and **c**[9] correspond, respectively, to the

posterior-most position on the abdomen and the anterior-most position on the head. All other control points are symmetric along the anteroposterior axis of the body (e.g., control points **c**[3] and **c**[15]). **(B)** Six leg anchor positions (yellow) between the coxa and thorax are defined empirically based on a linear combination of distances from the head-thorax boundary, the thorax-abdomen boundary, and a distance from the thoracic midline. These positions are then shifted depending on how the body model is optimally deformed to fit the contours of a specific animal. **(C)** The leg model consists of four control points including a thorax-coxa attachment **l**[0], the femur-tibia joint **l**[1], the tibia-tarsus joint **l**[2], and the pretarsus/claw **l**[3]. For simplicity, control points for only a single leg are shown. **(D)** In sum, 27 positions are calculated for each fly per frame: a centroid (0), anterior point (A), posterior point (P), as well as the body anchor, first intermediate, second intermediate and tip for each of the six legs. Our data labeling convention is as follows. Right and left legs are numbered 1 to 3 (front to rear) and 4 to 6 (front to rear), respectively. Each leg has four control points labeled 1 to 4 in the units digit that correspond the body anchor (1), leg joints (2 and 3), and claw (4). In each label, the leg number is shown in the tenths digit and the control point in the units digit. For example, the label "11" refers to the body anchor of the right prothoracic leg 1. For simplicity, only the control points for leg 3 are shown. **(E)** An example raw image of the ventral surface of a fly used for segmentation. **(F)** This image is first segmented using the parametric body snake consisting of 18 control points (red and blue crosses). **(G)** Subsequently, leg segmentation is initialized through automatic tracing from body anchor points to user-defined leg tips. From this initialization, an annotation is performed using open snakes consisting of four control points (yellow crosses). (**H**) Body and (**I**) leg segment tracking annotation for flies during a 455-frame (1.93 s) sequence. Annotation results (red) and the centroid in **H** or leg tip positions in **I** (blue) for each frame are overlaid.

https://doi.org/10.1371/journal.pone.0173433.g001

thorax-abdomen junction and the thorax length (Fig 1B). These reference locations are then adapted in accordance with deformations of the body model across the image sequence.

User input is required to initialize the positions of each leg prior to tracking. Initialization is based on a single click for each leg: the user indicates the claw (hereafter referred to as *tip*) of each leg through mouse-clicks on the selected frame. The click location is assigned to the most likely body anchor using a probabilistic formulation based on the distance and intersection with the fly's body model and that of other leg models. Once a leg tip and a leg anchor have been paired, a dynamic programming method [30] is initiated to automatically trace the leg from the anchor to the tip. To facilitate this process, the fly's legs are enhanced by processing the segmented image frame using a ridge detector [31].

Dynamic programming is a method that yields the globally optimal solution for a given separable problem. In particular, it can be used to implement algorithms solving shortest path problems. Dynamic programming relies on a graph-based representation: the shortest path is represented as a sequence of successive nodes in a graph that minimize a cost function. To trace a leg from its anchor to its tip, we build a graph by interpolating image pixels along the two orthonormal axes. The first axis (axis **k**, indexed by k) is given by the unit vector along the straight segment linking the anchor of a leg to its tip. The second axis (axis **u**, indexed by u) is perpendicular to the first axis and is thus given by the normal vector to **k**. On a path, we refer to as u_k the index u corresponding to a given k. A path is therefore described by a collection of nodes (k, u_k). The cost of the path at index $k + 1$ along axis **k** is then given by:

$$C[k+1] = C[k] + \lambda\left(\frac{1}{L_S}\sum\nolimits_{(x,y)\in S} I_{\mathrm{ridge}}(x,y)\right) + (1-\lambda)|u_k - u_{k+1}|, \tag{11}$$

where $C[i]$ is the cost of the path at location i on axis **k**, S is the collection of image pixels (x,y) in the segment between nodes (k, u_k) and ($k + 1$, u_{k+1}), L_S is the pixel length of the considered segment, I_{ridge} is the ridge-filtered version of the current frame, and $\lambda \in [0,1]$ is a weighting coefficient. The first term corresponds to a discretized integral of the image in the segment linking nodes k and $k + 1$, and therefore tends to favor paths going through low pixel values. The second term is composed of the distance along axis **u** between two successive nodes and thus prevents large jumps along the **u** axis. As a result, the optimal path follows relatively bright (or dark) regions in the image with respect to the background in accordance with the first term in Eq 11, while retaining a certain level of smoothness due to the second term. The relative contributions of each term are determined by λ.

In contrast to body segmentation, leg segmentation uses open rather than closed snakes. Fly legs are parameterized by a curve composed of $M = 4$ control points (Fig 1C and 1G). For each leg, the body anchor, $\mathbf{l}[0]$, is considered fixed. The discrete path obtained through dynamic programming is used to initialize the leg snake. The rationale behind this two-step procedure is two-fold. First, dynamic programming is very robust and can therefore effectively trace the leg from a body anchor to its tip. However, since it is a discrete approach, it is computationally expensive. By contrast, snake-based methods are more likely to diverge when initialized far from their target but are computationally inexpensive since only a few control points need to be stored to characterize a given curve. Therefore, we combined these approaches by first finding a path to define each leg using dynamic programming and then transforming this path into a parametric curve for optimization. The parametric representation of the leg snake curve is defined as

$$\mathbf{s}(t) = \begin{pmatrix} s_1(t) \\ s_2(t) \end{pmatrix} = \sum_{k=0}^{M-1} \mathbf{l}[k]\varphi(Mt - k), \tag{12}$$

where $t \in [0,1]$ and $\mathbf{l}[k] = \{(l_1[k]\ l_2[k])^{\mathrm{T}}\}_{k\in\mathbb{Z}}$ are the leg snake control points. Since *Drosophila* legs are composed of relatively straight segments between each joint, we use linear splines as basis functions $\varphi(t)$. The leg control points are therefore linked through linear interpolation and each control point has a unique identifier that can be used for subsequent data processing (Fig 1D). Fig 1E–1G illustrates the full process of taking a single raw image (Fig 1E) and using active contours to segment the body (Fig 1F) and legs (Fig 1G).

Segmentation propagation (tracking). High frame-rate videos ensure that the displacement of a fly's body between successive frames is small. FlyLimbTracker takes advantage of this fact to propagate body and leg snakes from one frame to the next during tracking. The body snake in frame *t+1* is therefore segmented by optimizing a contour initialized as the corresponding snake from frame *t* using the body snake energy previously described. This approach is sufficient to obtain good segmentation provided that there is some overlap between the animal's body in frames *t* and *t+1*.

Compared with the body, leg displacement can be larger between frames. Therefore, leg snakes require a more sophisticated algorithm to be propagated during tracking. First, the anchor of each leg is automatically computed from the newly propagated fly body. Since each leg is modeled as a 4-node snake, the three remaining leg snake control points are optimized using the snake energy

$$E_{\mathrm{leg}} = E_{\mathrm{ridge}} + E_{\mathrm{EDT}} + E_{\mathrm{segments}} + E_{\mathrm{extremity}}. \tag{13}$$

The first term corresponds to the integral along the leg in the current frame filtered by a ridge detector [31], i.e.,

$$E_{\mathrm{ridge}} = \int_C I_{\mathrm{ridge}}\,\mathrm{d}s = \int_0^1 I_{\mathrm{ridge}}(\mathbf{s}(t))|\mathbf{s}'(t)|\mathrm{d}t, \tag{14}$$

where I_{ridge} is the ridge-filtered version of the current frame and $\mathbf{s}(t)$ is the snake curve as described by Eq 12.

Analogous with the first term, the second term is computed as the integral along the leg of the Euclidean distance transform (EDT, [32]) in the current frame where

$$E_{\mathrm{EDT}} = \int_C I_{\mathrm{EDT}}\,\mathrm{d}s = \int_0^1 I_{\mathrm{EDT}}(\mathbf{s}(t))|\mathbf{s}'(t)|\mathrm{d}t, \tag{15}$$

where I_{EDT} is the ridge-filtered version of the current frame and $\mathbf{s}(t)$ is the snake curve as described by Eq 12.

Each of the linear segments comprising a fly's legs should be roughly constant in length across a video, aside from changes introduced by projecting the three-dimensional legs onto two-dimensional images. Taking this consistency into account, the third term of the leg energy penalizes solutions for which the leg joint positions result in leg segments whose lengths vary considerably from one frame to the next. This prevents unrealistic configurations of the leg joints that yield excessively long leg segments compared with neighboring annotated frames. Formally, E_{segments} corresponds to the sum of absolute differences in length between each segment of the target leg at frames t and $t + 1$.

Finally, the fourth term is used to determine the leg tip position at time t, denoted $\mathbf{l}_t[3]$ because it corresponds to the third control point of the leg snake (Eq 12). Since the distal tip of the leg may move considerably between successive frames, we designed a dedicated energy term to attract the tip toward candidate locations in the image. These candidate locations are defined by minima after the image is filtered using a Laplacian-of-Gaussian (LoG, [33]). A potential map P of points $\mathbf{p} = (p_x, p_y)$ that are tip candidates is then created according to:

$$E_{\text{extremity}} = 1 - w_{\mathbf{p}^*} \mathrm{e}^{-\frac{\|\mathbf{l}_t[3]-\mathbf{p}^*\|^4}{\sigma^2}}, \tag{16}$$

where

$$\mathbf{p}^* = \operatorname{argmin}_{\mathbf{p}\in P} \|\mathbf{l}_t[3] - \mathbf{p}\|^2 \tag{17}$$

is the tip candidate closest to $\mathbf{l}_t[3]$, $w_{\mathbf{p}^*} \in [0, 1]$ is its associated weight, and σ^2 is a fixed parameter determining the width of the attraction potential of the tip candidates. The weight $w_{\mathbf{p}^*}$ is a measure of how tip-like $\mathbf{p}^*$ is, and is computed based on the magnitude of the LoG filter response. A strong weight results in a deeper potential, and is therefore more likely to attract $\mathbf{l}_t[3]$.

In summary, the four anchor points characterizing each leg are propagated as follows. First, the leg body anchors are determined using the body model. Second, the remaining three control points (two leg joints and tip) are shifted by optimizing a cost function that incorporates both image information (E_{ridge} and E_{EDT}) and a temporal smoothness constraint (E_{segments}). Finally, the tip is further constrained using an estimate of how tip-like the image is at candidate locations.

Data output. Once the full image sequence is annotated, data can be extracted as a CSV file for each fly. These measurements include the locations of three reference points on the fly's body (A, P, and 0), as well as each of the legs' anchor points (see Fig 1D for the labeling convention).

FlyLimbTracker is linked to Icy's Track Manager plug-in (Publication Id: ICY-N9W5B7) via the *extract tracks* buttons (see interface description in the Appendix), allowing additional data to be extracted. In particular, segmentations of the fly's body (Fig 1H) and legs (Fig 1I) can be visualized across the entire sequence, illustrating their entire trajectories. Each individual control point of the leg snakes or the body snake's centroid can be independently visualized. Note that tracks are also numbered according to the labeling convention in Fig 1D.

Tracking multiple flies. FlyLimbTracker can track multiple flies in a single field of view. Additional flies are marked and tracked in a similar way as the first one. Then, the tracking algorithm relies on a multithread implementation to avoid increased processing time. Since the location of the tracked flies is manually determined by user clicks in the first frame, the presence of other objects in the field of view will not disturb the tracking algorithm as long as they do not occlude the selected flies. If selected flies are occluded, the user can use a manual mode to annotate frames. Note that the design of the fly body model (Fig 1A) dictates that each fly is at least 10 pixels long and 8 pixels wide. The quality of segmentation and tracking strongly depends on resolution because higher resolution images contain more information. However, segmentation of large images also requires more computer memory and is thus slower. This trade-off is further investigated and discussed in the Results section.

Cross-platform compatibility. Because the most memory-intensive step of the algorithm is image loading, FlyLimbTracker can be used on any computer capable of opening relevant image sequences. Performance times are not expected to be strongly dependent on the operating system since the software is implemented in Java, a multiplatform language. Processing times reported in the Results section of this manuscript can thus be used as a reference for comparison.

Software and data availability

User instructions, FlyLimbTracker software, sample data, and a video demonstration of a complete data analysis pipeline can be found at:

http://bigwww.epfl.ch/algorithms/FlyLimbTracker/

and at

https://doi.org/10.6084/m9.figshare.4688962.v1

Results

FlyLimbTracker performs semi-automated body and leg tracking. First, the user manually initializes the positions of the fly's body and leg segments in a single, arbitrarily chosen frame of the image sequence (Fig 2A). These manual annotations are then used to automatically propagate segmentation to prior, or subsequent frames (Fig 2B). During automated segmentation, the user can interrupt tracking to correct errors (Fig 2C). When FlyLimbTracker is restarted, the automated segmentation continues, taking into account these user edits.

Algorithm robustness

FlyLimbTracker can be used to segment and track fly bodies and legs in videos spanning a wide range of spatial and temporal resolutions. Resolution determines the nature of the annotation process: high-resolution data tracking is more automated, while low resolution data requires more user intervention. To quantify the dependence of computing time and the number of user interventions on data quality, we systematically varied the spatial and temporal resolutions of videos featuring five common *Drosophila* behaviors: walking straight (3 walking cycles using a tripod gait), turning (>90° turn), foreleg grooming (3 leg rubs), head grooming (3 head rubs), and abdominal grooming (3 abdominal rubs). These five videos were derived from two longer movies: one movie of a fly walking straight and grooming its forelegs, and another movie of a different fly turning, grooming its head, and grooming its abdomen. Raw videos were originally captured at 236 fps and a resolution of 2560 x 918 pixels (S1–S5 Videos).

First, we studied FlyLimbTracker's robustness to variations in spatial resolution. To cleanly isolate the effects of spatial resolution, rather than acquire different data with lower resolution cameras, we down-sampled each of the original five videos by a factor of N, where $N \times N$ pixels were averaged. This resulted in image sequences N times smaller along both spatial dimensions but with an identical temporal resolution of 236 fps (Fig 3A). Alternatively, to vary temporal resolution, we down-sampled each video by a factor of N, where only one frame from every N

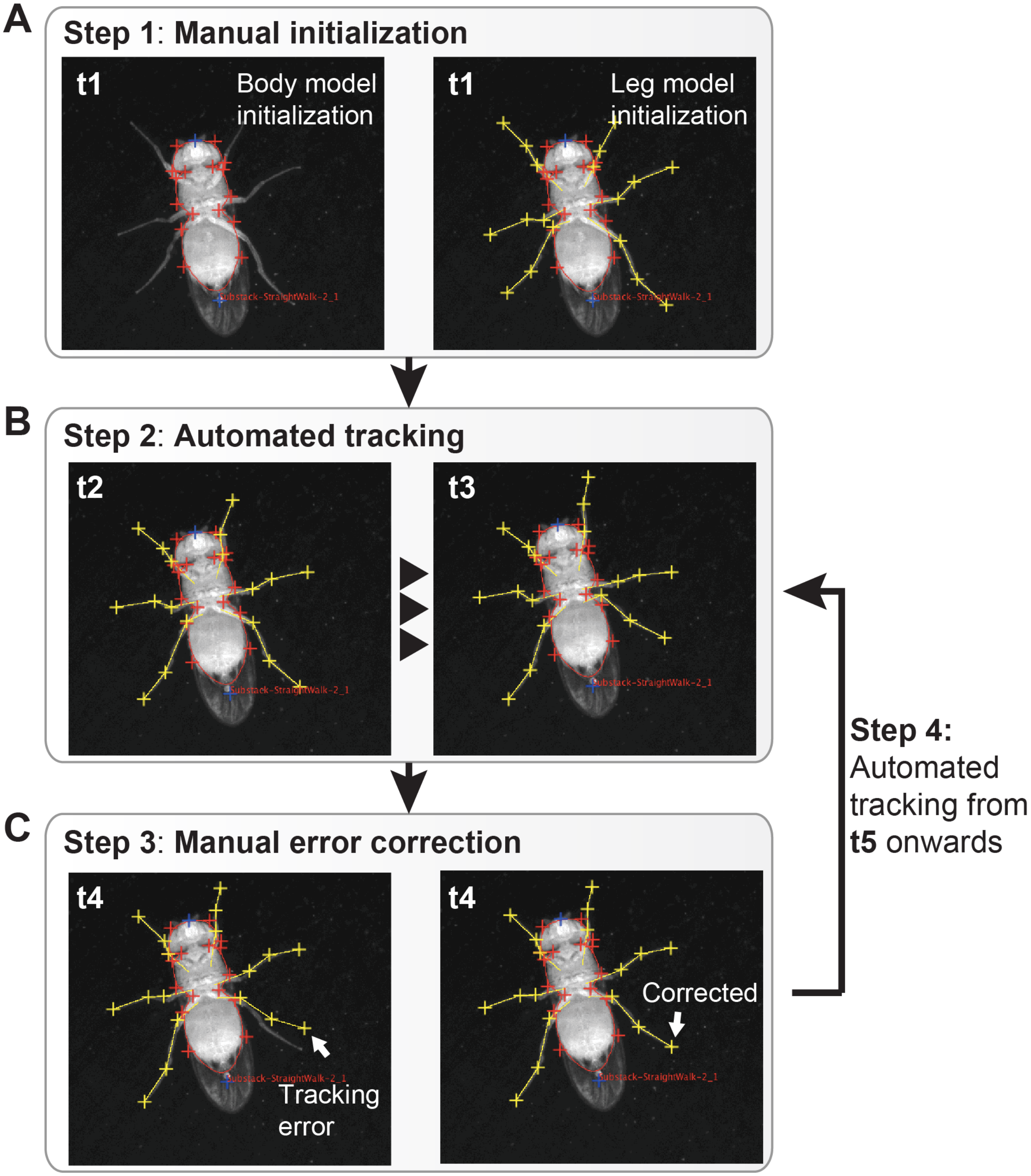
A
Step 1: Manual initialization
t1
Body model initialization
t1
Leg model initialization
Substack-StraightWalk-2_1
B
Step 2: Automated tracking
t2
t3
C
Step 3: Manual error correction
t4
Tracking error
t4
Corrected
Step 4: Automated tracking from t5 onwards

Fig 2. FlyLimbTracker workflow. (A) The user manually indicates the approximate location of the fly's body in an arbitrarily chosen video frame (t1). FlyLimbTracker then optimizes a closed active contour model that encapsulates the fly's body in the correct orientation. The user then manually indicates the location of each leg's tip. FlyLimbTracker then optimizes an open active contour model that runs across the entirety of each leg. **(B)** The user then runs FlyLimbTracker's automatic tracking algorithm to propagate body and leg models to subsequent video frames (or prior frames if run in reverse). **(C)** Either during or after automated tracking, the user can look for tracking errors. After manually correcting these errors, the user can re-run automatic tracking. In each image, the frame number is indicated.

https://doi.org/10.1371/journal.pone.0173433.g002

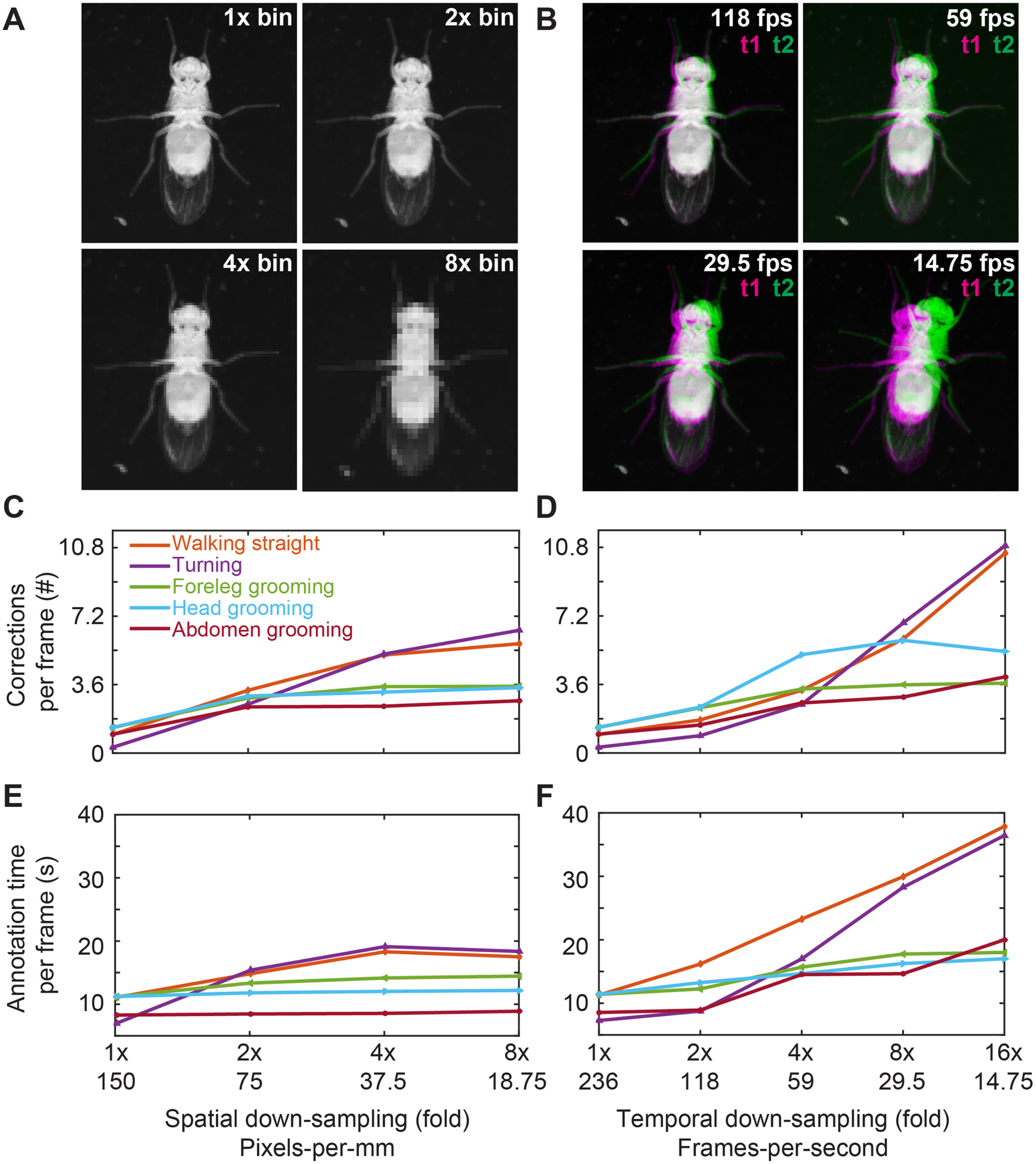

Fig 3. Sensitivity of leg tracking to changes in spatial or temporal video resolution. (A) Sample video image (top-left) after 2x (top-right), 4x (bottom-left), or 8x (bottom-right) spatial down-sampling. Adult female flies imaged are approximately 375, 187, 93, and 46 pixels in length in the 1x, 2x, 4x, and 8x

spatial down-sampled videos, respectively. **(B)** Representations of the difference between successive images (t1 and t2 overlaid in magenta and green, respectively) for different frame rate videos after temporal down-sampling. **(C-D)** The number of corrections required per frame as a function of spatial resolution **(C)**, or temporal resolution **(D)**. **(E-F)** The average time required to semi-automatically annotate a single frame as a function of spatial resolution **(E)**, or temporal resolution **(F)**. In C-F, data for videos depicting a fly walking straight, turning, grooming its forelegs, head, or abdomen are shown in orange, purple, green, cyan, and red, respectively.

https://doi.org/10.1371/journal.pone.0173433.g003

was retained. This resulted in image sequences of varying temporal resolution but consistently high spatial resolution of 2560 x 918 pixels (Fig 3B).

For each movie, body and leg snakes were manually initialized using the first image frame. Segmentation was then automatically propagated forward through the remainder of the image sequence. Whenever the automated tracker made a mistake, the process was interrupted and the user manually corrected the error. Automated tracking was then restarted from this frame until the next mistake was observed. In all cases, automated body tracking did not require manual intervention. Therefore, we only took note of manual corrections in leg snake annotation.

The throughput of image annotation using FlyLimbTracker can be determined by comparing the software with fully manual annotation. Rather than time spent annotating–a metric that can vary dramatically between users–we quantify the number of required mouse clicks. Fully manual annotation of four control points for six legs, and one click to advance to the next frame is 25 mouse clicks per frame (24 clicks for the final frame). By contrast, FlyLimbTracker requires initialization for the first frame (four clicks to optimize the body model and at least one click to initialize each leg model) and, in the worst-case scenario (head grooming), less than two corrections per frame with high spatial and temporal resolution (Fig 3C, far left, 1x Spatial down-sampling). By conservatively assuming that there is an error in each frame, we add two mouse clicks per frame to stop and then restart tracking. Therefore, using FlyLimbTracker we can expect an average of four mouse clicks per frame. This is approximately a 6-fold increase in throughput when compared with fully manual annotation.

To quantify FlyLimbTracker's performance across this range of spatial and temporal resolutions, we calculated two normalized quantities. First, we calculated the average number of manual corrections per frame (Fig 3C and 3D). To do this, we measured the total number of user interventions while processing an image sequence and normalized this quantity by T, where T is the number of frames, each of which contains eighteen free parameters: six legs with three editable control points each. As a second metric we quantified the average time required to annotate a single image frame (Fig 3E and 3F). To do this, we recorded the total time required to annotate an image sequence and divided this value by the total number of frames. This normalized quantity combines both the computing time required for automated annotation as well as the time required to manually correct annotation errors. Overall, we observed that reducing spatial (Fig 3A, 3C and 3E), or temporal (Fig 3B, 3D and 3F) resolution resulted in an increase in the number of manual interventions (Fig 3C and 3D) as well as a longer time required for annotation (Fig 3E and 3F).

While the numbers of corrections were similar for equivalent amounts of down-sampling (up to 8-fold), annotation time was appreciably longer for straight walking and turning. This reflects the importance of having overlapping images in successive frames for automated tracking: a feature that may be less common during locomotion where the position of a leg can vary substantially within a walking cycle. Notably, in a number of other cases (e.g., grooming), the annotation time per frame flattens across spatial and temporal resolutions. This is probably due to the trade-off between resolution and speed. Resolution strongly influences the computing time required for automated tracking: smaller images or sequences composed of fewer frames are processed more quickly due to reduced demands on computer memory. However, a decrease in resolution also implies a reduction in the quantity of image information and an increase in the likelihood of image processing errors. More user intervention is therefore required to correct mistakes. These interventions begin to dominate the time required to annotate each frame. In summary, intermediate image resolutions are ideal for FlyLimbTracker since very low resolution images may require almost fully manual annotation while very high resolution image annotation can be prohibitively memory intensive.

Visualization and analysis of leg segment tracking data

FlyLimbTracker provides a user-friendly interface that allows body and leg segment tracking data to be exported in a CSV file format, simplifying data analysis and visualization. We illustrate three representations of body and leg tracking data for annotated videos of the five behaviors previously described (S6–S10 Videos). Data interpretation and the number of experiments required to test statistical significance are study/experiment-dependent and therefore beyond of the scope of this work.

First, within FlyLimbTracker itself, leg joint and/or body trajectories can be displayed overlaid upon the final raw video frame (Fig $4A_1$–$4E_1$) using the TrackManager Icy plug-in. This representation provides a way to project time-varying data onto a static image and illustrates the symmetric or asymmetric limb motions that control straight walking/grooming or turning, respectively. Second, leg segment trajectory data can be exported and processed externally using Matlab or Python. An example of such a script is provided on the FlyLimbTracker website. Second, these data can be rotated along with the fly's frame of reference (Fig $4A_2$–$4E_2$) for a direct comparison of leg segment movements across distinct actions. Third, joint and claw movements can be isolated (Fig $4A_3$–$4E_3$) to generate a visualization that is routinely used to show how genetic perturbations, or strain-dependent differences influence claw movements during locomotion [12,34]. FlyLimbTracker permits a similar representation to be used to also visualize previously inaccessible leg joints and new, non-locomotive behaviors (e.g., grooming or reaching). In a fourth visualization, the speeds of each claw can be plotted to provide an exceptionally detailed characterization of locomotor gaits (Fig $4A_4$–$4B_4$), or grooming movements in stationary animals (Fig $4C_4$–$4E_4$). These are just a few examples of how to analyze tracking data. Simple post-processing would also permit, for example, measurements of joint angles as well as the relative position of each joint with respect to any other annotated body part (e.g., head, thorax, or abdomen).

Discussion

Existing methods for tracking insect leg segments rely on sophisticated optical equipment and/or laboriously-applied leg markers, often in tethered animals [10–12]. While these approaches are extremely valuable, they may potentially disrupt natural behaviors and cannot report the motions of multiple joints in untethered animals. Here we have introduced a method using active contours and other computer vision techniques to address these technical barriers. The software implementation of this approach, FlyLimbTracker, permits semi-automated tracking of body and leg segments in freely behaving *Drosophila*. Use of FlyLimbTracker only requires a single high-resolution, high-speed camera and does not require prior marking of leg segments. Additionally, it can be used with video data across a range of spatial and temporal resolutions, permitting a flexible blend of automated and manual annotation. Importantly, when automation has difficulty segmenting low quality data, FlyLimbTracker remains a powerful tool for manual leg tracking annotation since it uses easily manipulated spline-snakes and provides an interface for user-friendly data import and export. Exported data–fly limb and body position in image coordinates–serve as the basis for computing a range

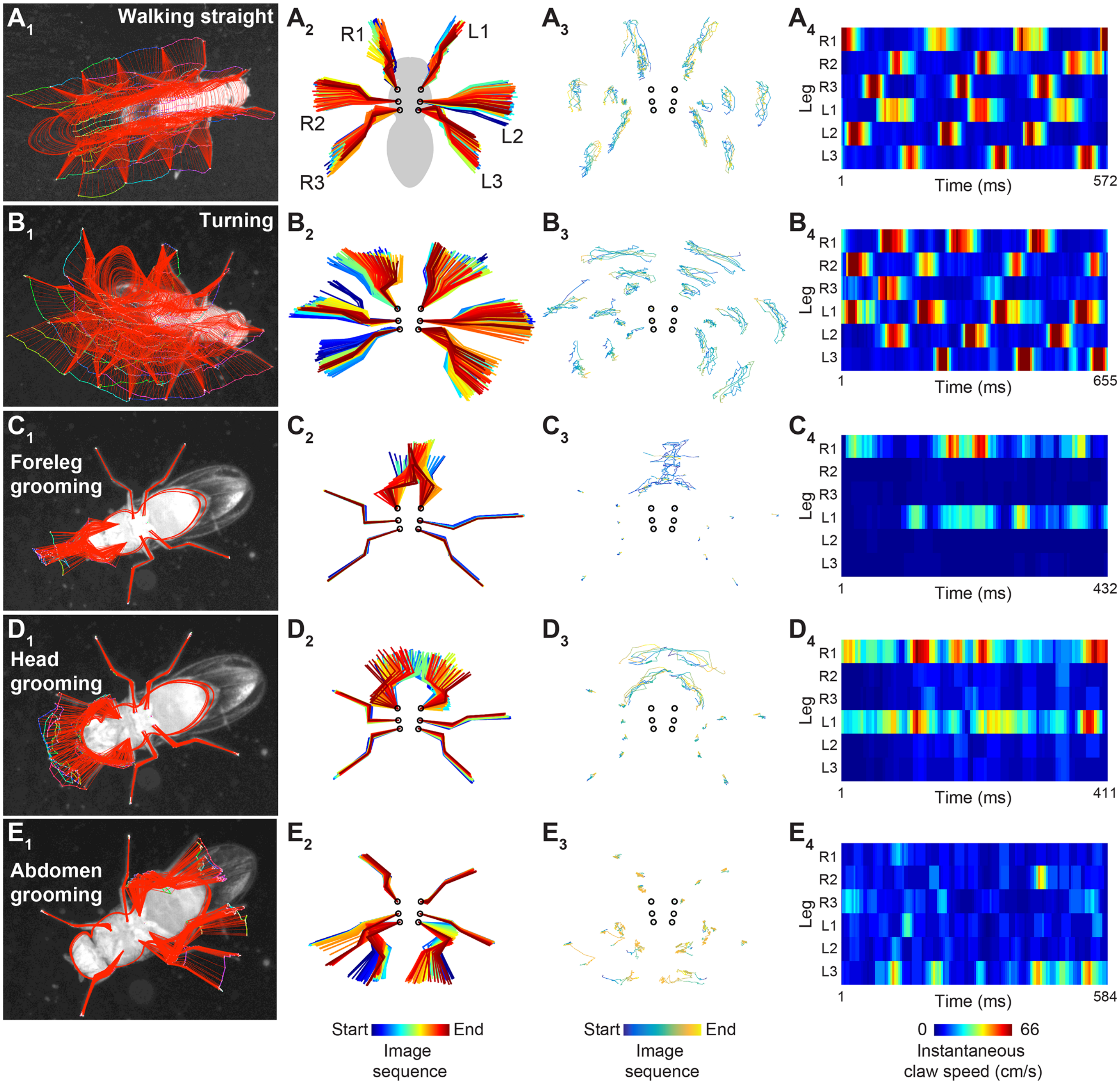

Fig 4. Analysis and visualization of FlyLimbTracker leg tracking data. Visualizations of leg segment annotation results for videos of a fly **(A)** walking straight, **(B)** turning, **(C)** grooming its forelegs, (**D**) grooming its head, or (**E**) grooming its abdomen. **(A_1-E_1)** Leg segmentation results (red) and joint positions (color-coded by frame number) are overlaid on the final frame of the image sequence. **(A_2-E_2)** Leg segment trajectories are rotated and color-coded by frame number. This permits alignment and comparison of leg movements across different datasets. **(A_3-E_3)** Joint and claw movements are represented in isolation. **(A_4-E_4)** The instantaneous speeds of each leg tip (claw) are color-coded.

https://doi.org/10.1371/journal.pone.0173433.g004

of statistics describing fly motion using custom software (e.g., Python or Matlab scripts). Of course, the exact statistics depend entirely on the experimental setting and biological problem under consideration.

Moving forward it will be important to discover how well active contour limb tracking functions in other contexts (e.g., low lighting, alternative camera angles, following limb removal, mutant animals with unpredictable limb motions). Although the number of possibilities is enormous, we can already make informed predictions about which algorithmic steps may be sensitive to specific experimental variations. First, active contours are general and flexible, allowing them to accommodate a variety of morphologies including male and female flies that are known to have different sizes and shapes. Second, if one or more limbs have been removed, our software has been implemented to allow for the annotation of only a subset of the fly legs. Third, the motion model used for leg tracking is linear and should provide best results in settings where fly legs move in a consistent direction between successive frames. However, we provide a slider on the FlyLimbTracker user interface that makes it possible to change the degree to which the tracking algorithm depends on the motion model as opposed to image information (see Appendix). Less reliance on this motion model would be advised when tracking mutant animals exhibiting unpredictable leg movements. FlyLimbTracker is currently sensitive to certain changes in experimental conditions. First, our fly segmentation model has been designed for experiments in which the fly is seen from above or below. Therefore, although small variations of viewing angle may be acceptable, side views cannot be accomodated. Second, accurate body tracking requires that the fly's body is overlapping between successive frames. Third, the algorithm is only robust to heterogeneities in lighting if they are present over the whole image sequence; Punctuated changes in image quality or lighting can disrupt tracking.

The open-source nature of FlyLimbTracker facilitates community-driven improvement and customization of the software. We envision a number of improvements that may be implemented moving forward. First, tracking currently requires overlap of a fly's body between successive frames. This constraint places a lower bound on video temporal resolution and could be improved by using, for example, nearest-neighbor matching approaches like the Hungarian algorithm [35] to link segmentation control points between successive frames. Second, additional leg control points may be added to FlyLimbTracker to more precisely annotate thorax-coxa-trochanter segments. Similarly, the current system with a single camera projects three-dimensional joint position data onto two-dimensions, making it difficult to accurately measure every joint angle. The segmentation additional camera views could permit full three-dimensional reconstruction of each limb's orientation and position in space. Third, FlyLimbTracker's requirement of user initialization, makes it only semi-automated and restricts batch processing of multiple videos for high-throughput data analysis. This may be overcome by using additional prior information to automatically identify and optimize body snakes. Fourth, FlyLimbTracker's snake-based approach to tracking could easily be adapted for the study of other species (e.g., mice, stick insects, and cockroaches) by modifying the shape of snake models.

Appendix

User interface

FlyLimbTracker's interface can be used in either basic or advanced mode. In the basic mode, only the name of the active image is visible. All parameters are hidden and only default parameter values are used. When switching to the advanced mode, all parameters become visible and can be adjusted by the user. Parameters that can be adjusted in the interface include:

- Image parameters
    - Channel: for multichannel images (e.g., bright-field and fluorescence), this parameter selects the channel upon which segmentation is performed. In most cases, the bright-field channel should be selected.

- ○ Smoothing: adjusts the width (standard deviation, in pixels) of a smoothing filter used to preprocess the image sequence. Larger values yield smoother images, but likely obscure details such as the fly's legs. We recommend choosing a value approximately equal to the average width (in pixels) of the fly legs.

- ○ Subtract background: performs background subtraction on the image sequence. The background model used is the median of each pixel across the whole image sequence. In practice, background subtraction is not desirable in datasets with a low signal-to-noise ratio since a fly's legs typically have low contrast and can be smoothed out by median filtering.

- Body model parameters

  - ○ Annotation method: switches between automated and manual annotation of the body snake. Automated annotation is obtained by automatically optimizing the body snake from its initial, manually chosen position. Manual annotation relies exclusively on user interactions.

  - ○ Energy trade-off: adapts the relative importance of data fidelity (image-based) and regularization (shape-based) terms in the body snake energy. A fully image-based snake would be optimized using image information only, while a fully shape-based snake would be optimized to retain a fly's shape regardless of the underlying image data. For data with low image quality the regularization term (shape-based) becomes more important.

  - ○ Max iterations/immortal: tunes the maximum number of iterations used to optimize the body snake. If *immortal* is chosen, the snake keeps evolving until it achieves convergence. Allowing the snake to be immortal usually yields better segmentation results, but significantly increases computing time. Conversely, a smaller number of iterations can estimate segmentation quickly, but not necessarily as effectively. Usually, 4000–5000 iterations provide a good trade-off between computing time and segmentation quality. However, this value should be customized according to data quality.

  - ○ Freeze snake body: when ticked, locks the control points of the fly body snake, which then appear as blue instead of red. In this setting, individual points cannot be further edited. This feature is useful when the fly body is properly initialized and edits are done on the legs only, as it prevents displacing body control points when trying to select a leg control point. However, it remains possible to translate, move or rotate the entire fly body.

- Leg model parameters

  - ○ Annotation method: switches between automated and manual segmentation of the fly's legs. Although body segmentation and tracking is robust even for low resolution or low signal-to-noise ratio data, leg tracking is much more sensitive. Therefore, the user is given the option to restrict automation to body tracking. In the manual segmentation setting, the legs are simply propagated by translation along with body motion and can be manually adjusted post-hoc for each frame. This allows FlyLimbTracker to be a useful tool for annotating either low-quality or high-quality data.

  - ○ DP trade-off: determines the relative importance of data fidelity (bright) and regularization (straight) terms when performing dynamic programming (DP) to initialize the leg snakes. The algorithm tries to find the optimal path between a given leg anchor and tip by optimizing the trade-off between image intensity (bright) and straightness (straight). Relying on image brightness alone typically yields irregular movements of the fly's legs since the algorithm becomes very sensitive to image noise (e.g., isolated pixels of high intensity). Conversely, relying on straightness alone yields, in the most extreme case, a straight line between the anchor and tip. Note that this parameter is only used when initializing a leg. It does not influence tracking.

- Energy trade-off: determines the relative importance of data fidelity (image-based) and regularization (sequence-based) terms for the leg snakes. A purely image-based leg snake is optimized using the image data only. This typically yields suboptimal solutions that are sensitive to image noise. Conversely, a fully sequence-based leg snake maximizes its resemblance to the corresponding leg snake from previously annotated frames and ignores image data. More importance should be given to sequence-based energy for low quality data when leg snake annotations are readily available.
- Tip propagation mode: determines the relative importance of data fidelity (image-based) and regularization (sequence-based) terms while tracking leg tips. We identify potential tips by searching for candidate locations in a neighborhood encompassing leg motions from previously annotated, neighboring frames. The final tip position is chosen as a trade-off between the position predicted by leg motion from previous annotated frames (sequence-based), and tip candidates identified by processing the current frame (image-based).
- Max iterations/immortal: tunes the maximum number of iterations used to optimize the leg snakes in a manner similar to how the same parameter is used to optimize the body snake.

In both basic and advanced modes, the upper part of the interface contains several menu items (Analyze, Save/Load and Help):

- Analyze: extracts measurements from the current body segmentation using Icy's ROI Statistics plug-in (Publication Id: ICY-W5T6J4).
- Save/Load: allows the user to export and save annotations to a CSV file format (see Output section below). This can also be used to reload previously saved CSV annotations.
- Help: contains information about the plug-in version (About), and a link to FlyLimbTracker's online documentation page (Documentation (online)).

Finally, several action buttons are located on the lower part of the interface. These are split into three sections.

- Fly shape editing: the left button enables movement of individual control points. The middle and right buttons, respectively, enable resizing and rotation of the body and leg snakes.
- Snake action: automatically optimizes the snake at its current position (left button), or deletes it (right button). Note that both actions are applied to the body snake and all leg snakes simultaneously. If annotation methods for body or leg snakes are set to *manual*, the corresponding snakes are left unmodified.
- Tracker action: performs backward (left button) or forward (center-left button) tracking, interrupts tracking (center-right button), or extracts/displays tracks (right button) using Icy's Track Manager plug-in (Publication Id: ICY-N9W5B7). The tracking algorithm is implemented to allow backward and forward tracking, giving the user flexibility to initialize tracking at any frame of the image sequence. If any of the body or leg snakes are set to manual annotation, the forward and backward tracking buttons will only propagate current

annotations to the next or previous frame, respectively. If all snakes are set to automated annotation, tracking will be performed in the selected direction until the end/beginning of the image sequence is reached, unless it is manually halted using the tracking interruption button.

Supporting information

S1 Video. A fly walking straight.
(MOV)

S2 Video. A fly turning.
(MOV)

S3 Video. A fly grooming its forelegs.
(MOV)

S4 Video. A fly grooming its head.
(MOV)

S5 Video. A fly grooming its abdomen.
(MOV)

S6 Video. A fly walking straight (video 1), annotated using FlyLimbTracker.
(MOV)

S7 Video. A fly turning (video 2), annotated using FlyLimbTracker.
(MOV)

S8 Video. A fly grooming its forelegs (video 3), annotated using FlyLimbTracker.
(MOV)

S9 Video. A fly grooming its head (video 4), annotated using FlyLimbTracker.
(MOV)

S10 Video. A fly grooming its abdomen (video 5), annotated using FlyLimbTracker.
(MOV)

Acknowledgments

We thank Cédric Vonesch, Michael Rusterholz, and Loic Perruchoud for early contributions to the body tracking algorithm.

Author Contributions

Conceptualization: PR VU MU.

Data curation: PR VU.

Formal analysis: VU RDG PR.

Funding acquisition: PR RB MU.

Investigation: PR VU.

Methodology: VU RDG MU.

Project administration: PR MU.

Resources: RB MU.

Software: VU RDG PR.

Supervision: PR RB MU.

Validation: VU RDG PR.

Visualization: PR VU.

Writing – original draft: PR VU.

Writing – review & editing: PR VU RDG RB MU.

References

1. Olsen SR, Wilson RI. Cracking neural circuits in a tiny brain: new approaches for understanding the neural circuitry of *Drosophila*. Trends Neurosci. 2008; 31: 512–520. https://doi.org/10.1016/j.tins.2008.07.006 PMID: 18775572
2. Noldus L, Spink AJ, Tegelenbosch R. Computerised video tracking, movement analysis and behaviour recognition in insects. Computers and Electronics in agriculture. 2002; 35: 201–227.
3. Dankert H, Wang L, Hoopfer E, Anderson DJ, Perona P. Automated monitoring and analysis of social behavior in *Drosophila*. Nature Methods. 2009; 6: 297–303. https://doi.org/10.1038/nmeth.1310 PMID: 19270697
4. Branson KM, Robie AA, Bender JA, Perona P, Dickinson MH. High-throughput ethomics in large groups of *Drosophila*. Nature Methods. Nature Publishing Group; 2009; 6: 451–457. https://doi.org/10.1038/nmeth.1328 PMID: 19412169
5. Donelson N, Kim EZ, Slawson JB, Vecsey CG, Huber R, Griffith LC. High-Resolution Positional Tracking for Long-Term Analysis of *Drosophila* Sleep and Locomotion Using the "Tracker" Program. PLoS One. 2012; 7: e37250. https://doi.org/10.1371/journal.pone.0037250 PMID: 22615954
6. Pérez-Escudero A, Vicente-Page J, Hinz RC, Arganda S, de Polavieja GG. idTracker: tracking individuals in a group by automatic identification of unmarked animals. Nature Methods. 2014; 11: 743–748. https://doi.org/10.1038/nmeth.2994 PMID: 24880877
7. Deng Y, Coen P, Sun M, Shaevitz JW. Efficient Multiple Object Tracking Using Mutually Repulsive Active Membranes. PLoS One. 2013; 8: e65769. https://doi.org/10.1371/journal.pone.0065769 PMID: 23799046
8. Berman GJ, Choi DM, Bialek W, Shaevitz JW. Mapping the stereotyped behaviour of freely moving fruit flies. Journal of The Royal Society Interface. 2014; 11: 20140672–20140672.
9. Berman GJ, Bialek W, Shaevitz JW. Predictability and hierarchy in Drosophila behavior. Proceedings of the National Academy of Sciences. 2016; 113: 11943–11948.
10. Kain J, Stokes C, Gaudry Q, Song X, Foley J, Wilson RI, et al. Leg-tracking and automated behavioural classification in *Drosophila*. Nature Communications. 2013; 4: 1910–1918. https://doi.org/10.1038/ncomms2908 PMID: 23715269
11. Bender JA, Simpson EM, Ritzmann RE. Computer-Assisted 3D Kinematic Analysis of All Leg Joints in Walking Insects. PLoS One. 2010; 5: e13617. https://doi.org/10.1371/journal.pone.0013617 PMID: 21049024
12. Mendes CS, Bartos I, Akay T, Márka S, Mann RS. Quantification of gait parameters in freely walking wild type and sensory deprived *Drosophila melanogaster*. eLife. 2013; 2.
13. Isakov A, Buchanan SM, Sullivan B. Recovery of locomotion after injury in Drosophila melanogaster depends on proprioception. Journal of . . .. 2016.
14. Pick S, Strauss R. Goal-Driven Behavioral Adaptations in Gap-Climbing *Drosophila*. Current Biology. 2005; 15: 1473–1478. https://doi.org/10.1016/j.cub.2005.07.022 PMID: 16111941
15. Seeds AM, Ravbar P, Chung P, Hampel S, Midgley FM, Mensh BD, et al. A suppression hierarchy among competing motor programs drives sequential grooming in *Drosophila*. eLife. 2014; 3.
16. de Chaumont F, Dallongeville S, Chenouard N, Hervé N, Pop S, Provoost T, et al. Icy: an open bioimage informatics platform for extended reproducible research. Nature Methods. 2012; 9: 690–696. https://doi.org/10.1038/nmeth.2075 PMID: 22743774
17. de Chaumont F, Coura RD-S, Serreau P, Cressant A, Chabout J, Granon S, et al. Computerized video analysis of social interactions in mice. Nature Methods. 2012; 9: 410–417. https://doi.org/10.1038/nmeth.1924 PMID: 22388289

18. Chenouard N, Buisson J, Bloch I, Bastin P, Olivo-Marin J-C. Curvelet analysis of kymograph for tracking bi-directional particles in fluorescence microscopy images. IEEE 17th International Conference on Image Processing. 2010;: 3657–3660.

19. Delgado-Gonzalo R, Uhlmann V, Schmitter D, Unser M. Snakes on a Plane: A perfect snap for bio-image analysis. IEEE Signal Process Mag. 2015; 32: 41–48.

20. Dénervaud N, Becker J. A chemostat array enables the spatio-temporal analysis of the yeast proteome. 2013. pp. 15842–15847.

21. Schmitter D, Wachowicz P, Sage D, Chasapi A, Xenarios I, Simanis V, et al. A 2D/3D image analysis system to track fluorescently labeled structures in rod-shaped cells: application to measure spindle pole asymmetry during mitosis. Cell Division. 2013; 8.

22. Kass M, Witkin A, Terzopoulos D. Snakes: Active contour models. International journal of computer vision. 1988;: 321–331.

23. Delgado-Gonzalo R, Chenouard N, Unser M. Spline-Based Deforming Ellipsoids for Interactive 3D Bio-image Segmentation. IEEE Transactions on Image Processing. 2013; 22: 3926–3940. https://doi.org/10.1109/TIP.2013.2264680 PMID: 23708807

24. Brigger P, Hoeg J, Unser M. B-spline snakes: a flexible tool for parametric contour detection. IEEE Transactions on Image Processing. 2000; 9: 1484–1496. https://doi.org/10.1109/83.862624 PMID: 18262987

25. Delgado-Gonzalo R, Thévenaz P, Unser M. Computer Aided Geometric Design. Computer Aided Geometric Design. Elsevier B.V; 2012; 29: 109–128.

26. Delgado-Gonzalo R, Thévenaz P, Seelamantula CS, Unser M. Snakes With an Ellipse-Reproducing Property. IEEE Transactions on Image Processing. 2012; 21: 1258–1271. https://doi.org/10.1109/TIP.2011.2169975 PMID: 21965208

27. Jacob M, Blu T, Unser M. Efficient Energies and Algorithms for Parametric Snakes. IEEE Transactions on Image Processing. 2004; 13: 1231–1244. PMID: 15449585

28. Delgado-Gonzalo R, Schmitter D, Uhlmann V, Unser M. Efficient Shape Priors for Spline-Based Snakes. IEEE Transactions on Image Processing. 2015; 24: 3915–3926. https://doi.org/10.1109/TIP.2015.2457335 PMID: 26353353

29. Press WH, Flannery BP, Teukolsky SA, Vetterling WT. Numerical recipes: the art of scientific computing. Cambridge University Press; 1986.

30. Dijkstra EW. A note on two problems in connexion with graphs. Numerische mathematik. 1959;: 269–271.

31. Jacob M, Unser M. Design of steerable filters for feature detection using canny-like criteria. IEEE Transactions on Pattern Analysis and Machine Intelligence. 2004; 26: 1007–1019. https://doi.org/10.1109/TPAMI.2004.44 PMID: 15641731

32. Felzenszwalb PF, Huttenlocher DP. Distance Transforms of Sampled Functions. Theory of Computing. 2012; 8: 415–428.

33. Sage D, Neumann FR, Hediger F, Gasser SM, Unser M. Automatic tracking of individual fluorescence particles: application to the study of chromosome dynamics. IEEE Transactions on Image Processing. 2005; 14: 1372–1383. PMID: 16190472

34. Wosnitza A, Bockemühl T, Dübbert M, Scholz H, Büschges A. Inter-leg coordination in the control of walking speed in *Drosophila*. The Journal of Experimental biology. 2013; 216: 480–491. https://doi.org/10.1242/jeb.078139 PMID: 23038731

35. Kuhn HW. The Hungarian method for the assignment problem. Naval research logistics quarterly. 1955;: 83–97.